SEFRON: A New Spiking Neuron Model With Time-Varying Synaptic Efficacy Function for Pattern Classification

Jeyasothy Abeegithan, Sundaram Suresh, and Narasimhan Sundararajan

***Abstract*—This paper presents a new time-varying long-term Synaptic Efficacy Function based leaky-integrate-fire neuRON model, referred to as SEFRON and its supervised learning rule for pattern classification problems. The time-varying synaptic efficacy function is represented by a sum of amplitude modulated Gaussian distribution functions located at different times. For a given pattern, the SEFRON's learning rule determines the changes in the amplitudes of weights at selected presynaptic spike times by minimizing a new error function reflecting the differences between the desired and actual postsynaptic firing times. Similar to the GABA (Gamma-Aminobutyric Acid) -switch phenomenon observed in a biological neuron which switches between excitatory and inhibitory postsynaptic potentials based on the physiological needs, the time-varying synapse model proposed in this paper allows the synaptic efficacy (weight) to switch signs in a continuous manner. The computational power and the functioning of SEFRON are first illustrated using a binary pattern classification problem. Detailed performance comparisons of a single SEFRON classifier with other spiking neural networks are also presented using four benchmark datasets from the UCI machine learning repository. The results clearly indicate that a single SEFRON provides similar generalization performance compared to other spiking neural networks with multiple layers and multiple neurons.**

***Index Terms*—spiking neurons, time-varying synaptic efficacy function, SEFRON, STDP, GABA-switch.**

I. Introduction

In recent times, Spiking Neural Networks (SNNs) are being developed with increasing interest because of their biologically relevant functionalities and also the high computational power that they possess, as compared to sigmoidal neural networks. Spiking neural networks require a lower number of neurons compared to a sigmoidal neural network for approximating the same function, implying that a SNN is computationally more powerful than a sigmoidal neural network of the same size [1]. From the machine learning point of view, the core research activities in spiking neural networks have been in the areas of developing efficient supervised learning algorithms like SpikeProp [2], Synaptic Weight Association Training (SWAT) [3], ReSuMe [4], Tempotron [5], others [6]–[13]. The main objectives of these algorithms have been learning the functional relationships between the input and output spike patterns. Learning algorithms in spiking neural networks employ different neuron models such as Hodgkin-Huxley model [14], Leaky-Integrate-and-Fire (LIF) model [15], [16] or Spike Response Model (SRM) [17], [18]. In literature, the most commonly used spiking neuron model for the development of SNNs is the LIF model or its equivalent SRM because of their simpler forms and ease of developing learning algorithms. The SRM uses a kernel integration scheme to determine the postsynaptic potential. A LIF neuron model can be mapped into a SRM where the excitatory and inhibitory postsynaptic potentials are defined as products of spike response functions and synaptic efficacies [19].

These neuron models provide the flexibility to access the synaptic efficacy directly and develop learning algorithms to adjust it without affecting other neuron properties. In SNNs, two spiking neurons are connected via a synapse model. A synapse model represents the strength of a connection between two spiking neurons. The strength of the connection between i^{th} input neuron and j^{th} output neuron is characterized by its weight w_{ij}. This weight determines the amplitude of the postsynaptic response (typically indicated by the height of the postsynaptic potential or the slope of the postsynaptic current). The synaptic plasticity/synapse models have been normally overlooked during the development of earlier supervised learning algorithms in SNNs [2], [4]–[13] as most of them use only the long-term plasticity models (constant weight) [20]–[22].

Based on current research in neuroscience, biological synaptic plasticity models can be broadly classified into two categories, viz., homosynaptic plasticity or heterosynaptic plasticity. In homosynaptic plasticity models, the properties of the synapses are modified by their internal activities (activities of the neurons that are connected by the same synapses) and in heterosynaptic plasticity models, the properties of the synapses are modified by external activities (eg., modulatory substances such as acetylcholine, dopamine, histamine, serotonin etc.). Heterosynaptic plasticity models are beyond the scope of the work of this paper and are not highlighted here. The main focus here is only on the dynamic models of homosynaptic plasticity that are mainly used in SNNs, viz., that of short-term plasticity models [23]–[28]. Static homosynaptic plasticity models will also not be discussed further here as it has been pointed out in [23], [29] that a spiking neural network with a dynamic plasticity has more computational power than a SNN with only a static plasticity (long-term plasticity).

Short-term plasticity models include the release of neurotransmitters [23] in synaptic connections. Both facilitation [26] and depression [24], [25] can also be modelled in short-term plasticity models. These short-term plasticity models can be

Jeyasothy Abeegithan (abeegith1@e.ntu.edu.sg), Sundaram Suresh (ssundaram@ntu.edu.sg) and Narasimhan Sundararajan(ensundara@ntu.edu.sg) are with the School of Computer Science and Engineering, Nanyang Technological University, Singapore

interpreted as neural connections with event-driven dynamic weights. But the weights in this model recover to constant values in the absence of spiking activity and is same as that of a long-term plasticity model with constant weights.

In a supervised learning framework, incorporating short-term plasticity models has been a challenging task. A SNN with a dynamic synapse model in [23] was developed for speech recognition task and it showed that a simple 2-layer network can perform well for the selected dataset. However, this system failed when applied to a large database [30]. Further improvements to that network's architecture and algorithm to estimate the parameters of the dynamic synapses were proposed in [31], [32]. Recently, a spiking neural network for pattern classification called SWAT [3], which uses a dynamic synapse model [26] along with a long-term plasticity model was presented. However, from the results presented for SWAT, it can be observed that the resulting network is large and involves a high computational load. Besides the challenges in implementing short-term plasticity, it is also important to note that the short-term plasticity models can only be employed on top of constant weights (long-term plasticity).

Instead of implementing the biological short-term plasticity models on top of long-term plasticity models, in this paper, we propose a new method to model a dynamic synapse suitable for SNNs by distributing the long-term plasticity over a specific time window (interval). With this objective, we present a time-varying synaptic efficacy function ($w_{\text{ij}}(t)$) as the weight instead of a constant synaptic weight (w_{ij}) to model a synapse. The synaptic efficacy function is defined as a summation of different amplitude modulated Gaussian distribution functions with their centers located at different times in the time window. In this paper, using this newly defined Synaptic Efficacy Function, along with a leaky integrate and fire neuRON model is referred to as SEFRON. This SEFRON model is then used for developing the learning algorithm for SNNs. This $w_{\text{ij}}(t)$ approximates the amplitude variation required in a weight within a time window based on both the presynaptic and postsynaptic activities for all the training patterns in that time window. Hence, the final weight of a single synaptic connection is a continuous time-varying function that is independent of incoming presynaptic spikes during the time window.

Recently, in the neuroscience literature, an interesting switching phenomenon has been observed in a synapse. This mechanism is such that it switches the same synapse from an excitatory nature to inhibitory nature and vice versa [33], [34]. This switching phenomenon has been referred to as the GABA (gamma-aminobutyric acid) –switch. This phenomenon has been observed during the development of a human brain. In an infant's brain, GABA receptor-mediated responses are producing excitatory postsynaptic potentials. During the development of the adult brain (exclusively) it has been observed that the GABA receptor-mediated postsynaptic potential changes from excitatory to inhibitory nature [33]. Also, experiments conducted on lactating rats showed that the GABA receptor-mediated responses in oxytocin and vasopressin neurons are converted back into excitatory in a reversible manner [34]. These studies show that the GABA-switch occurs when there is a physiological need. Inspired by this phenomenon, in this paper, we have not restricted the proposed time-varying synaptic model ($w_{\text{ij}}(t)$) in SEFRON to be either excitatory or inhibitory nature, i.e, the weight is allowed to change in a continuous manner from a positive value to a negative value and vice versa during the specified time window.

For pattern classification problems, using this SEFRON model, we also present a supervised learning algorithm to determine the $w_{\text{ij}}(t)$ using a normalized form of the standard STDP rule. For a given pattern, the supervised learning rule first calculates V_{STDP}, the postsynaptic potential due to the fractional contributions (or in other words normalized form of STDP). Then, it computes the change in the weight (amplitude of the $w_{\text{ij}}(t)$) by minimizing an error function based on the desired and actual postsynaptic spike times. This error function e is represented by the difference in the ratios of the firing threshold (θ) to the V_{STDP} of selected presynaptic spikes for the desired and actual postsynaptic spike times. Next, the learning rule uses the computed weight change to update the $w_{\text{ij}}(t)$, by forming its product with a Gaussian distribution function centered at the selected presynaptic spike times. As a consequence of SEFRON's learning rule, $w_{\text{ij}}(t)$ for some synapses may have both positive and negative values within the time window, which imitates the GABA-switch phenomenon.

The computational power of a single SEFRON is then illustrated using a binary classification problem. Here, the class labels for both classes are coded into two different desired postsynaptic spike times ($\hat{t}_d^1, \hat{t}_d^2$). For classification purposes, the postsynaptic spike interval is split into two regions such that each of the desired postsynaptic spike times falls within only one of the regions. The splitting time is defined as the boundary spike time ($\hat{t}_b$). Classification decision is made by the occurrence of the actual postsynaptic spike ($\hat{t}_a$) with respect to the boundary spike time, i.e the class label is determined as class 1 if the postsynaptic spike occurs before the boundary spike time ($\hat{t}_a < \hat{t}_b$) and class 2 otherwise ($\hat{t}_a > \hat{t}_b$). Further, a skipping sample strategy similar to that was used in [35] has been employed to prevent overtraining. Detailed performance evaluation of SEFRON is presented using four of the UCI machine learning data sets and its performance is compared with four well-known SNN algorithms viz.SpikeProp [2], SWAT [3], OSNN [9] and SRESN [7]. Based on the results, it can be seen that a single SEFRON outperforms all the other online learning SNN algorithms [7], [9]. The results also show that SEFRON produces similar accuracies as that of other offline learning SNN algorithms [2], [3] but with a single SEFRON.

The rest of the paper is organized as follows. The SEFRON model and its learning rule is highlighted in section II. The functioning of a single SEFRON as classifier is illustrated in section III. Detailed performance comparison of SEFRON with other existing learning algorithms is presented in section IV. Finally, the conclusions from this study are summarized in section V.

II. Development of SEFRON Model and its Learning Rule

In this section, the details of the proposed SEFRON model along with its learning algorithm are described. Existing

synapse models require different synapses to model both the excitatory and inhibitory characteristics resulting in a larger network to encode the data distribution on a feature space. However, it is shown that the proposed synapse model is able to encapsulate the data distribution with a single neuron.

A. Single SEFRON Model

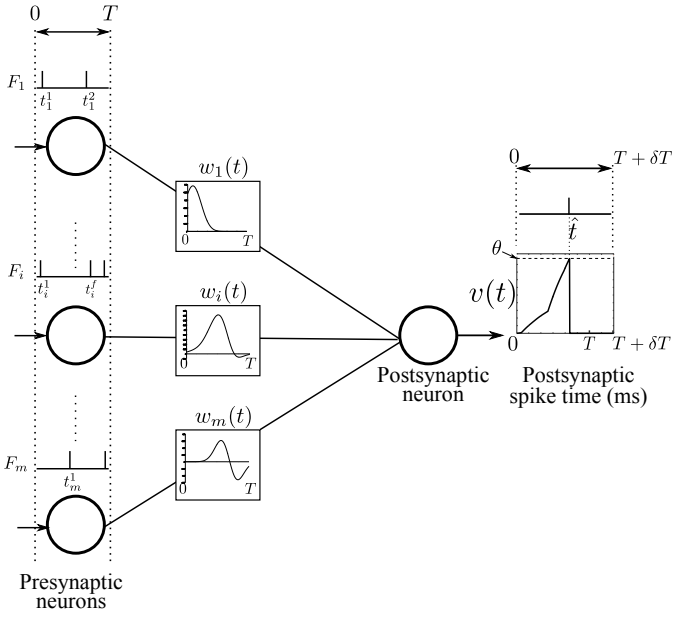


Fig. 1: A SEFRON model with m number of synapses. Presynaptic spike time for a given input pattern is in the interval of $[0, T]ms$ and postsynaptic spike time is in the interval of $[0, T + \delta T]ms$.

Figure 1 shows a single LIF neuron with the time-varying long-term synapse model (SEFRON). A set of input synapses $\Gamma = \{1 \text{ to } m\}$ are connected to a postsynaptic neuron. The set of presynaptic spike times F_i of the i^{th} synapse ($i \in \Gamma$) is defined as,

$$F_i = \{t_i^k; 1 \leqslant k \leqslant n_i\} \tag{1}$$

where n_i is the total number of presynaptic spikes fired by the i^{th} synapse and k represents the order of the firing. t_i^k is the firing time of the k^{th} presynaptic spike fired by the i^{th} input neuron. Let the presynaptic spike times for a given input pattern be in the time window (interval) $[0, T]ms$. The postsynaptic firing time is represented by $\hat{t}$ and is set in the interval $[0, T + \delta T]ms$ with the limit of postsynaptic spike time extended by δT to capture the late postsynaptic spikes and also to allow the SEFRON model to fire a postsynaptic spike at a later time.

Since a single postsynaptic neuron model is used in our work, the notation for the synaptic efficacy function between the i^{th} input neuron and the output neuron is denoted as $w_{i1}(t)$ but for simplicity, is written as $w_i(t)$. The postsynaptic potential $v(t)$ of SEFRON is defined as the summation of the product of the spike response of the presynaptic spike and the momentary weight determined by $w_i(t)$ for all presynaptic spikes (fired by all synapses). It is defined as,

$$v(t) = \sum_{i=1}^{m} \sum_{k=1}^{n_i} w_i(t_i^k).\epsilon(t - t_i^k) \tag{2}$$

where $\epsilon(t)$ is the spike response function [2] and is given as,

$$\epsilon(t) = \frac{t}{\tau} exp(1 - \frac{t}{\tau}) \tag{3}$$

τ is the time constant of LIF neuron and $w_i(t_i^k)$ refers to the momentary synaptic efficacy (weight) and is the value of $w_i(t)$ at $t = t_i^k$,

$$w_i(t_i^k) = \{w_i(t)|_{t=t_i^k}\} \tag{4}$$

SEFRON fires a postsynaptic spike when the postsynaptic potential reaches firing threshold θ. The postsynaptic firing time $\hat{t}$ is defined as,

$$\hat{t} = \{t | v(t) = \theta\} \tag{5}$$

where,

$$\theta = v(\hat{t}) = \sum_{i=1}^{m} \sum_{k=1}^{n_i} w_i(t_i^k).\epsilon(\hat{t} - t_i^k) \tag{6}$$

As $w_i(t)$ is a time-varying function of synaptic efficacy with respect to presynaptic spike time, the interval for $w_i(t)$ is also set to $[0, T]ms$ to coincide with the interval of presynaptic spike times.

B. SEFRON's Learning Rule

The principles behind SEFRON's learning rule are briefly explained below. First, the learning rule computes $V_{\text{STDP}}(\hat{t})$ using the fractional contribution of presynaptic spikes resulting in the 'actual' postsynaptic spike time and also computes the same for the 'desired' postsynaptic spike time. Next, it computes the change in the synaptic efficacy by minimizing an error function which represents the difference in the ratio of θ to $V_{\text{STDP}}(\hat{t})$ at the desired and actual postsynaptic spike times. Finally, the learning rule modulates the change in synaptic efficacy using a Gaussian distribution function centered at the current presynaptic spike time.

A normalized form of STDP based rule is used [12] to compute the fractional contribution of presynaptic spikes for a given postsynaptic spike. The general STDP learning rule defines the synaptic efficacy change $\delta w(s)$ for a delay s, between the presynaptic and postsynaptic firing time as,

$$\delta w(s) = \begin{cases} +A_+.exp(-s/\tau_+) & \text{if s} \geqslant 0 \\ -A_-.exp(s/\tau_-) & \text{if s} < 0 \end{cases} \tag{7}$$

Here (A_+, τ_+) and (A_-, τ_-) are the maximum values of weight changes and plasticity window for the long term potentiation and long term depression respectively. Since only one postsynaptic spike is used in SEFRON to determine the fractional contributions, the presynaptic spikes fired after the first postsynaptic spike are ignored. For example, in pattern classification problems the first postsynaptic spike is important. Hence, other postsynaptic spikes are ignored. Therefore A_- is assumed to be 0 in SEFRON's learning rule. The fractional contribution $u_i^k(\hat{t})$ by presynaptic spike t_i^k to fire a postsynaptic spike at $\hat{t}$ is the normalized STDP (as in [12]) and is calculated as,

$$u_i^k(\hat{t}) = \frac{\delta w(\hat{t} - t_i^k)}{\sum_{i=1}^{m} \sum_{k=1}^{n_i} \delta w(\hat{t} - t_i^k)} \tag{8}$$

The term $u_i^k(\hat{t})$ is independent of variable A_+ and depends only on the plasticity window τ_+. Note that, the sum of all the fractional contributions $u_i^k(\hat{t})$ is equal to 1.

$$\sum_{i=1}^{m}\sum_{k=1}^{n_i} u_i^k(\hat{t}) = 1 \tag{9}$$

Fractional contributions can also be used to measure the importance of presynaptic spikes to fire a postsynaptic spike at any specific time. A higher value for fractional contribution $u_i^k(\hat{t})$ indicates that presynaptic spike at time t_i^k is more important than other presynaptic spikes for firing the postsynaptic spike at time $\hat{t}$. It can be noted that presynaptic spikes closer to a postsynaptic spike will have higher fractional contribution value compared to presynaptic spikes that are further away from a postsynaptic spike.

The postsynaptic potential due to fractional contribution $V_{\text{STDP}}(\hat{t})$ can be interpreted as the ideal postsynaptic potential at the time of firing if the STDP rule is employed in determining the weight. The $V_{\text{STDP}}(\hat{t})$ at time $\hat{t}$ is determined by replacing the momentary weight $w_i(t_i^k)$ with fractional contribution $u_i^k(\hat{t})$ in equation 6.

$$V_{\text{STDP}}(\hat{t}) = \sum_{i=1}^{m}\sum_{k=1}^{n_i} u_i^k(\hat{t}).\epsilon(\hat{t} - t_i^k) \tag{10}$$

But this $V_{\text{STDP}}(\hat{t})$ may not be always equal to the firing threshold (θ) (except in the ideal case) due to variations in the input data. Therefore the ratio of firing threshold θ to $V_{\text{STDP}}(\hat{t})$ is used as the measure to determine the overall strength ($\gamma_{\hat{t}}$) required by all the synapses to make the firing possible at time $\hat{t}$ as in equation 10.

$$\theta = \gamma_{\hat{t}}.V_{\text{STDP}}(\hat{t}) = \gamma_{\hat{t}}.\sum_{i=1}^{m}\sum_{k=1}^{n_i} u_i^k(\hat{t}).\epsilon(\hat{t} - t_i^k) \tag{11}$$

where the overall strength $\gamma_{\hat{t}}$ is calculated as,

$$\gamma_{\hat{t}} = \frac{\theta}{V_{\text{STDP}}(\hat{t})} \tag{12}$$

In a supervised learning framework, a desired output is given and compared with the actual output. The weight is then adjusted to correct the differences between the desired output and the actual output. Here the desired and actual outputs are the desired postsynaptic spike time ($\hat{t}_d$) and the actual postsynaptic spike time ($\hat{t}_a$) respectively. Instead of directly computing the differences in the postsynaptic spike times, we have defined an error function that can compute the differences in the overall strength due to the differences in desired and actual postsynaptic spike times.

$$e = \gamma_{\hat{t}_d} - \gamma_{\hat{t}_a} = \frac{\theta}{V_{\text{STDP}}(\hat{t}_d)} - \frac{\theta}{V_{\text{STDP}}(\hat{t}_a)} \tag{13}$$

where $\gamma_{\hat{t}_d}$ and $\gamma_{\hat{t}_a}$ are the overall strengths due to the desired and actual postsynaptic spike times respectively.

This error e can be directly used to determine the change in weight. It can be noted by comparing equation 6 and equation 11 that the ideal momentary weight is the product of overall strength and fractional contribution ($w_i(t_i^k) = \gamma_{\hat{t}}.u_i^k(\hat{t})$ for the ideal case.). Multiplication of the error e in the overall strength with the fractional contribution for the desired postsynaptic spike time ensures that the actual momentary weight moves in a direction towards the ideal momentary weight. This ensures that the actual postsynaptic spike time ($\hat{t}_a$) moves towards the desired postsynaptic spike time ($\hat{t}_d$). The individual change in the synaptic efficacy $\triangle w_i(t_i^k)$ for i^{th} synapse at a presynaptic spike time t_i^k is calculated by multiplying the error in overall strength with the fractional contribution for the desired postsynaptic spike time.

$$\triangle w_i(t_i^k) = \lambda.u_i^k(\hat{t}_d).e \tag{14}$$

where λ is the learning rate and usually set to a smaller value. Equation 14 can be expanded as,

$$\triangle w_i(t_i^k) = \lambda.(\gamma_{\hat{t}_d}.u_i^k(\hat{t}_d) - \gamma_{\hat{t}_a}.u_i^k(\hat{t}_d)) \tag{15}$$

where $\gamma_{\hat{t}_d}.u_i^k(\hat{t}_d)$ is the ideal momentary weight. The other term $\gamma_{\hat{t}_a}.u_i^k(\hat{t}_d)$ is calculated from the actual postsynaptic spike time for a given pattern and it is not equal to the actual (current) momentary weight $w_i(t_i^k)$.

The change in synaptic efficacy computed here is a single value for a given input pattern. For other patterns that are similar to the current pattern, the synaptic efficacies should be similar. Hence, the current value of the synaptic efficacy is embedded in a time-varying function (a modulating function) that produces weights that are similar to the current one if the presynaptic spikes are nearer. In this study, a Gaussian distribution function is chosen as the modulating function. $\Delta w_i(t_i^k)$ at single time instance t_i^k is embedded in a time-varying function $g_i^k(t)$ as,

$$g_i^k(t) = \triangle w_i(t_i^k).exp(\frac{-(t - t_i^k)^2}{2\sigma^2}) \tag{16}$$

where σ is the efficacy update range. A smaller value for σ would capture more variations in the synaptic efficacy and an infinite value for σ would result in a constant $g_i^k(t)$ that resembles a long-term plasticity model.

Each synapse fires multiple presynaptic spikes. A time-varying synaptic efficacy function $w_i(t)$ for each synapse is obtained by adding all the changes in synaptic efficacy in an interval due to the different presynaptic spike times. The synaptic efficacy function update rule for i^{th} synapse is,

$$w_{\text{i new}}(t) = w_{\text{i old}}(t) + \sum_{k=1}^{n_i} g_i^k(t) \tag{17}$$

It can be noted that the updated $w_{\text{i new}}(t)$ may have both positive and negative values within the simulation interval. This imitates the GABA-switch phenomenon observed in a biological neuron.

III. SEFRON FOR PATTERN CLASSIFICATION PROBLEM

In this section, we illustrate the functioning of a single SEFRON for a binary pattern classification problem using a simple synthetic two-class linearly separable problem. In general, the input features of pattern classification problem are real-valued and they have to be converted first into spike patterns. Here, the normalized input data x_i ($x_i \in [0, 1]$) is

encoded into a spike pattern using the well known population encoding scheme given in [2]. In the population encoding scheme, x_i is projected into multiple receptive field neurons to generate presynaptic spikes. Each receptive field neuron generates only one presynaptic spike. The total number of receptive field neurons (q) determine the total number of spikes that will be generated for a given input value. Each receptive field neuron h ($h \in [1, q]$) has a firing strength ϕ_i^h for the input data x_i and it is computed as,

$$\phi_i^h = exp(-\frac{(x_i - \mu_h)^2}{2\sigma_{\text{pop}}^2}) \tag{18}$$

where μ_h and σ_{pop} are the center and the standard deviation of the h^{th} receptive field neuron respectively. μ_h and σ_{pop} are selected as in [2],

$$\mu_h = \frac{(2h-3)}{2.(q-2)} \tag{19}$$

$$\sigma_{\text{pop}} = \frac{1}{\beta.(q-2)} \tag{20}$$

where β is the overlap constant [2].

The firing time of each presynaptic spike is,

$$t_{\text{i\_h}}^1 = T \times [1 - \phi_i^h]. \tag{21}$$

here T is the limit of the presynaptic spike time interval.

A. SEFRON classifier

For a two-class problem, the class labels are coded into desired postsynaptic firing times as $\hat{t}_d^1$ for class-1 and $\hat{t}_d^2$ for class-2. Let $\hat{t}_b$ be the classification boundary referred to as the boundary spike time and it should satisfy the condition, $\hat{t}_d^1 < \hat{t}_b < \hat{t}_d^2$. For classification, only the first actual postsynaptic spike $\hat{t}_a$ is used. Thus, any postsynaptic spikes fired after the first postsynaptic spike is ignored. If SEFRON does not fire any postsynaptic spike, the firing time of the postsynaptic spike is taken as the end of the simulation time.

Initialization: Here, the first training pattern is used to initialize the $w_i(t)$ and θ. The ratio of threshold to postsynaptic potential due to the fractional contribution ($\frac{\theta}{V_{\text{STDP}}(\hat{t}_d)}$ or $\gamma_{\hat{t}_d}$) of the first sample is assumed to be 1. Hence θ is set to the value of $V_{\text{STDP}}(\hat{t})_{\hat{t}=\hat{t}_d}$ using equation 10 and 11.

$$\theta := \sum_{i=1}^{m} \sum_{k=1}^{n_i} u_i^k(\hat{t}_d).\epsilon(\hat{t}_d - t_i^k) \tag{22}$$

If the actual postsynaptic firing time is same as the desired firing time $\hat{t}_d$, then the following condition must be satisfied (see equation 6).

$$\theta = \sum_{i=1}^{m} \sum_{k=1}^{n_i} w_i(t_i^k).\epsilon(\hat{t}_d - t_i^k) \tag{23}$$

Hence, the initial momentary weight $w_i(t_i^k)$ of each synapse at the corresponding presynaptic spike time must be equal to $u_i^k(\hat{t}_d)$. The initial momentary weight is distributed using a Gaussian distribution function as,

$$w_{\text{i initial}}(t) = \sum_{k=1}^{n_i} u_i^k(\hat{t}_d).exp(\frac{-(t - t_i^k)^2}{2\sigma^2}) \tag{24}$$

In SEFRON, a sample is correctly classified if the postsynaptic spike is within the desired firing range (that is either fired after or before $\hat{t}_b$). The correctly classified samples are not used for updating the $w_i(t)$. Avoiding the samples that do not add new knowledge during the training phase improves the generalization on the class wise data distribution [35]. For other samples, the SEFRON learning algorithm described in equation 8 to 17 is used.

B. Synthetic binary classification problem

A linearly separable two class synthetic classification problem is considered. It consists of two variables ($x_1,\ x_2$) belonging to two classes as given below:

$$Class\ 1 = \begin{cases} 0 \leqslant x_1 \leqslant 0.4 \\ 0 \leqslant x_2 \leqslant 0.4 \end{cases} \quad Class\ 2 = \begin{cases} 0.6 \leqslant x_1 \leqslant 1 \\ 0.6 \leqslant x_2 \leqslant 1 \end{cases}$$

100 random samples (50 from each class) are generated for building the training and testing sets. For the population encoding scheme, the total number of receptive field neurons q is set to 6 and the overlap constant β is set to 0.7 as in [7]. The total number of input presynaptic neurons in SEFRON is determined by the product of number of receptive field neurons (q) and the dimension (m) of the input data. For this synthetic problem, the dimension of the input data m is 2. Hence the total number of input neurons is 12. A bias presynaptic neuron is also added and set to fire at $t = 0s$ to ensure that the postsynaptic potential of all the inputs start at $t = 0s$.

The presynaptic spike interval limits T is set to 3ms. Thus, the interval for $w(t)$ is also set to $[0, 3]ms$ to coincide with the spike interval. The postsynaptic spike interval is set to $[0, 4]ms$ (simulation time) to capture the late output spikes and also to model SEFRON to fire at a later time. The time constant of the spike response function has to be greater than the spike time interval for better convergence [2]. Hence the time constant τ for spike response function $\epsilon(t)$ is set to $3ms$. Desired spike firing times are the coded output labels for supervised learning. It has to be chosen within the simulation interval and also remain well separated. Hence, the desired firing time for $c1$ ($\hat{t}_d^1$) is set to the middle value of simulation time ($2ms$) and for $c2$ ($\hat{t}_d^2$) it is set to the end value of simulation time ($4ms$). These are the following four algorithm dependent parameters; the efficacy update range (σ), the plasticity window for STDP learning (τ_+), the learning rate (λ) and the boundary spike time ($\hat{t}_b$) and these are set as $\sigma = 0.5ms,\ \tau_+ = 0.6ms,\ \hat{t}_b = 3ms,\ \lambda = 0.5$ respectively.

For the performance evaluation, experiments were conducted in MATLAB R2015b using a 64-bit Windows 7 operating system in a CPU with 6 cores, 16 GB memory and 3.2GHz speed. SEFRON achieved the performance of 100% classification accuracy for both the training and testing data sets. For this problem, functioning of SEFRON is described by selecting one training sample from class 1-$[x_1 = 0.3790, x_2 = 0.0217]$ and class 2-$[x_1 = 0.6041, x_2 = 0.6887]$. The encoded spike patterns for these real valued $c1$ and $c2$ samples are [1.90, 0.68, 0.01, 0.64, 1.87, 2.67, 0.25, 0.13, 1.17, 2.29, 2.84, 2.98]ms and [2.64, 1.79, 0.57, 0.02, 0.76, 1.97, 2.79, 2.15, 0.97, 0.06, 0.39, 1.59]ms respectively. These 12 encoded spike times are

considered as the presynaptic firing times of the 12 input neurons. The times of the presynaptic spikes fired by all the input neurons for both classes and the final $w(t)$ obtained for all the 12 synapses at the end of the training are shown in figure 2.

In figure 2a-2d, the switching (in sign) similar to that of GABA-switch phenomenon can be observed in $w(t)$ indicating that those synapses can provoke both EPSP and IPSP for different presynaptic spike times. $w(t)$ in figure 2e-2j are always positive and therefore they would only provoke an EPSP. On the other hand, in figure 2k and 2l, $w(t)$ are always negative and would only provoke an IPSP. Due to the presence of the switching phenomenon in $w(t)$ in figure 2b-2d, the weights at the presynaptic spike times corresponding to $c1$ and $c2$ samples are positive and negative respectively, thereby provoking an EPSP for $c1$ sample and IPSP for $c2$ sample.

The postsynaptic potential $v(t)$ for both input patterns along with the presynaptic/postsynaptic spikes are given in figure 3. For the $c1$ sample, the postsynaptic potential $v(t)$ crosses θ at $1.92ms$, resulting in a postsynaptic spike at the same time. But, the $c2$ sample $v(t)$ does not reach θ, therefore there is no postsynaptic spike and hence it is assumed to occur at the end of simulation interval. Due to the presence of the switching phenomenon in some synapses, it can be observed that the postsynaptic potential for $c1$ input pattern is more positive when compared to $c2$ input pattern.

C. Guidelines for choosing the parameter values

The effects of the parameter setting on the performance of SEFRON were studied by changing one parameter at a time.

Effects of efficacy update range (σ): Efficacy update range determines the effect of the weight change in the presynaptic spike time interval. A smaller value of σ captures more variation in the weight and a larger value will result in minimal variation. Figure 4 shows the effect of sigma on the accuracy. From figure 4, it can observed that the value for σ between $0.05ms$ and $0.55ms$ gives the best performance for SEFRON. It is also seen that, for $\sigma \geqslant 1.5ms$, the performance of SEFRON is nearly constant and very low. For this σ value, the functioning of SEFRON is similar to that of a LIF neuron with a fixed weight. The σ was set at 0.5ms.

Effects of STDP learning window (τ_+): Figure 5 show the effect of τ_+ on the performance of SEFRON for the synthetic problem. The STDP learning window determines the contribution of each presynaptic spike for the change in weight. The contributions of the presynaptic spikes that fired much earlier to the postsynaptic spike increase and that fired closer to the postsynaptic spike decrease with increasing τ_+. From the figure 5, it can be seen that $\tau_+ > 0.35ms$ gives the best performance. Here, τ_+ was set as $0.6ms$.

Based on these studies, the general guidelines for selecting the most dominant parameters of SEFRON learning rule σ and τ_+ can be given as follows: $0.05ms \leqslant \sigma \leqslant 0.55ms$ and $\tau_+ > 0.35ms$. Two other hyper parameters, namely the boundary spike time $\hat{t}_b$ and the learning rate λ are problem dependent and are chosen appropriately using cross validation. Typically, choosing a middle value between the desired spike times for both classes ($\hat{t}_d^1$,$\hat{t}_d^2$) is a good choice for $\hat{t}_b$. The learning rate is normally set at a smaller value lower than 1. A higher value may leads to oscillation in the weights.

IV. Performance Evaluation of SEFRON

The performance of the SEFRON classifier has been evaluated using four benchmark data sets from the UCI machine learning repository and compared with two offline learning algorithms (SpikeProp [2], SWAT [3]) and two online learning algorithms (SRESN [7] and OSNN [9]). Details of the training and testing sets, number of features and the number of classes used for the performance comparison are given in table I. For each dataset, 10 random trial sets were generated for both the training and testing datasets to perform a 10-fold validation.

TABLE I: Description of Dataset used for validation

Dataset	# Features	# Classes	# Samples	
			Training	Testing
WBC	9	2	350	333
Ionosphere	33	2	175	176
PIMA	8	2	384	384
Liver	6	2	170	175

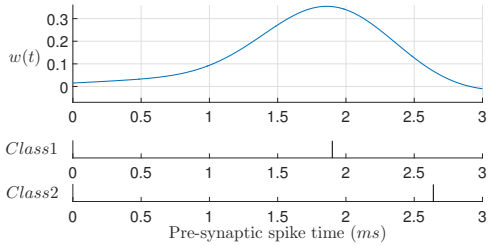
For each dataset, based on the guidelines given in section III-C, the values for the learning window, efficacy update range, boundary spike time and learning rate were determined. Table II shows the selected values for these four parameters for each dataset. Number of receptive field neurons in the population encoding scheme is set to 6, same as given in [7] to maintain the consistency in the representation of the datasets. All the other LIF neuron parameters and computing platform are kept same for all the studies and are given in section III-B.

TABLE II: Parameter values chosen for each data set

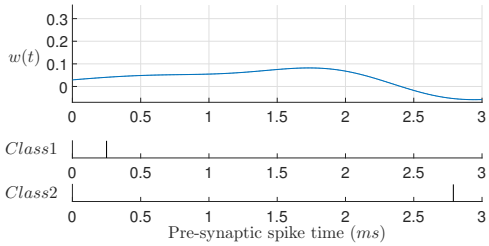
Data set	τ_+	σ	$\hat{t}_b$	λ
WBC	0.60	0.05	2.5	0.1
Ionosphere	0.55	0.15	3.0	0.5
PIMA	0.60	0.15	3.0	0.1
Liver	0.60	0.10	2.5	0.1

Results for all the other algorithms except for OSNN were generated using the same training and testing data sets, whereas the results for the OSNN has been reproduced from [9]. For SpikeProp, parameters were chosen following the guidelines given in [2] and 16 delayed terminals were used in the experimental study. Note that the number of neurons in the hidden layer is crucial in SpikeProp and it is determined by a constructive-destructive procedure as given in [36]. Since the population encoding scheme was used to generate the spike patterns, the parameter ISI in SWAT was set to $15 - 40ms$. The other parameters viz. the frequency filter array, c_0 and A_p were set to the same values as given in [3]. For SRESN, the same parameter settings given in [7] was used.

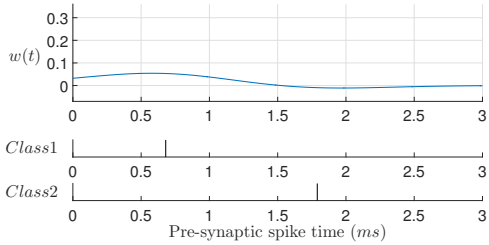
Based on the following four metrics; viz. the architecture (size) of the network, the training accuracy, the testing accuracy and the computation time, performance of SEFRON has been compared with the other methods. The architecture



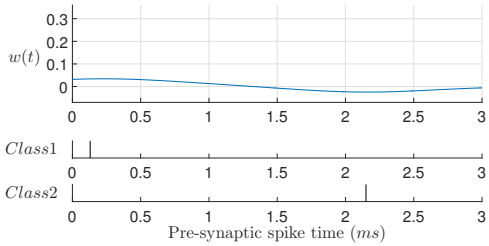
(a) Input neuron 1 of feature 1



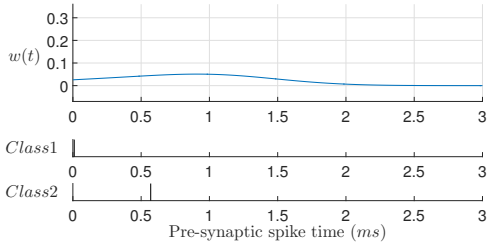
(b) Input neuron 1 of feature 2



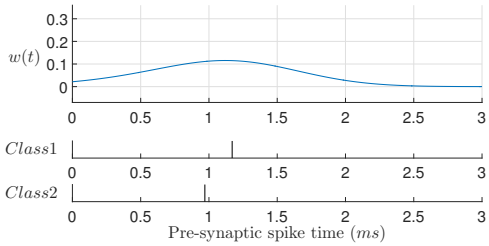
(c) Input neuron 2 of feature 1



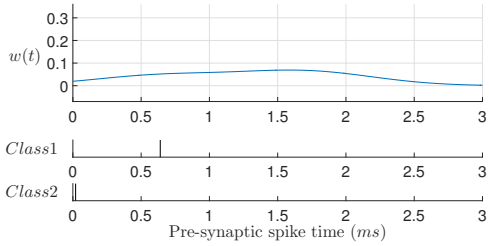
(d) Input neuron 2 of feature 2



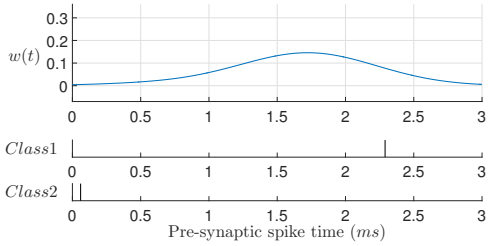
(e) Input neuron 3 of feature 1



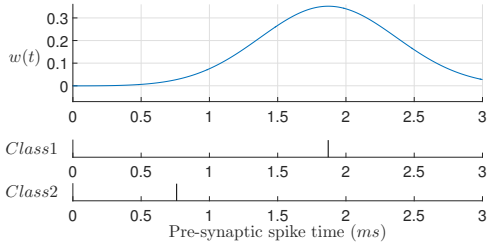
(f) Input neuron 3 of feature 2



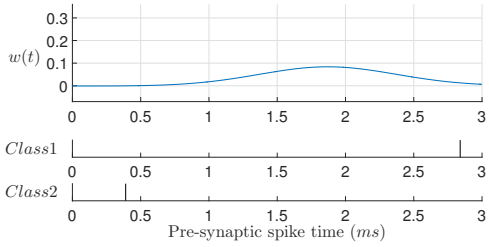
(g) Input neuron 4 of feature 1



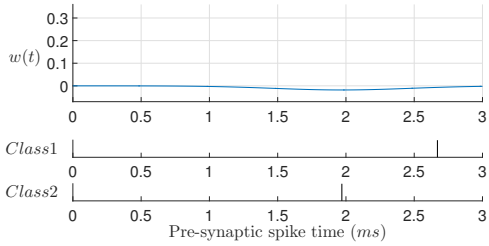
(h) Input neuron 4 of feature 2



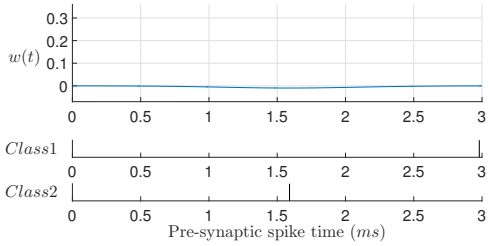
(i) Input neuron 5 of feature 1



(j) Input neuron 5 of feature 2



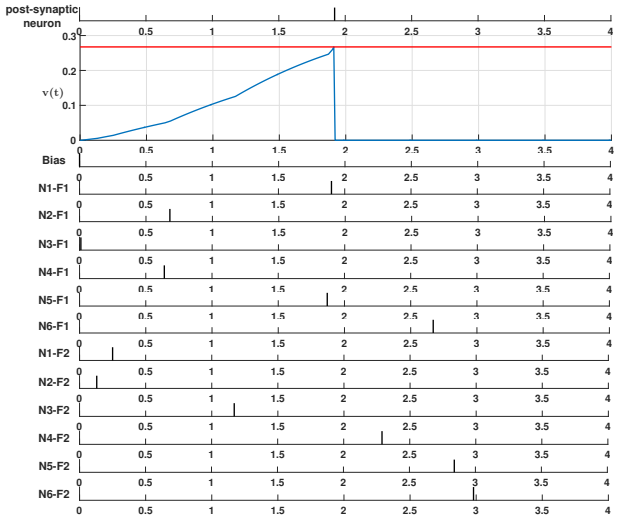
(k) Input neuron 6 of feature 1



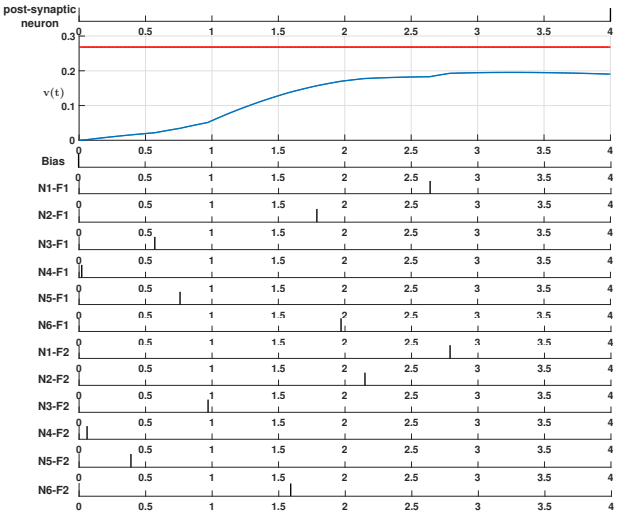
(l) Input neuron 6 of feature 2

Fig. 2: $w(t)$ of all the Input neurons and an example encoded input spike pattern for class 1 and class 2 sample.

of SNN is given in the form of $N_i : N_h : N_j$, where N_i, N_h and N_j are the total number of inputs, hidden



(a) PSP of class 1 sample and its presynaptic spike time.



(b) PSP of class 2 sample and its presynaptic spike time.

Fig. 3: PSP of $c1$ and $c2$ for the encoded presynaptic spike time. Here, for example N1-F1 refers to Input neuron 1 of feature 1

TABLE III: Performance comparison

Dataset	Learning Algorithm	Architechture	Training Accuracy (%)	Testing Accuracy (%)	Avg training Epoch time (s)	Max No.of Epoch
WBC	SpikeProp	55:15:2	97.3(0.6)	**97.2(0.6)**	3.75	1000
	SWAT	54:702:2	96.5(0.5)	95.8(1.0)	265.85	500
	OSNN	54:(10-16):2	91.1(2.0)	90.4(1.8)	-	1
	SRESN	54:(5-8)	93.9(1.8)	94.0(2.6)	5.24	1
	SEFRON	**55:1**	**98.3(0.8)**	**96.4(0.7)**	**0.48**	100
Ionosphere	SpikeProp	199:25:2	89.0(7.9)	86.5(7.2)	6.37	3000
	SWAT	198:2574:2	86.5(6.7)	**90.0(2.3)**	462.18	500
	OSNN	198:(4-11):2	76.7(2.4)	76.6(4.8)	-	1
	SRESN	198:(6-13)	85.1(1.9)	79.3(3.0)	9.43	1
	SEFRON	**199:1**	**97.0(2.5)**	**88.9(1.7)**	**0.45**	100
PIMA	SpikeProp	49:20:2	78.6(2.5)	**76.2(1.8)**	3.83	3000
	SWAT	48:624:2	77.0(2.1)	72.1(1.8)	253	500
	OSNN	48:(8-18):2	68.2(2.0)	63.5(3.0)	-	1
	SRESN	48:(6-12)	67.0(0.8)	66.1(1.4)	5.08	1
	SEFRON	**49:1**	**84.1(1.5)**	**74.0(1.2)**	**0.39**	100
Liver	SpikeProp	37:15:2	71.5(5.2)	65.1(4.7)	2.65	3000
	SWAT	36:468:2	74.8(2.1)	60.9(3.2)	83.17	500
	OSNN	36:(4-7):2	58.7(2.2)	56.7(1.8)	-	1
	SRESN	36:(5-8)	59.8(1.2)	57.4(1.1)	1.74	1
	SEFRON	**37:1**	**91.5(5.4)**	**67.7(1.3)**	**0.15**	100

and output neurons respectively. The computation time was calculated as the average time taken to complete one Epoch. Table III presents the performance comparison of SEFRON with other algorithms based on the four different data sets. The computation time for OSNN was not available, hence it is not reported in the table III.

From the Table III, it can be seen that for the WBC data set, the training and testing accuracies of all the methods are similar and also that they are higher as compared to the accuracies for other data sets. Since the WBC is a linearly separable problem, the classification accuracy of SEFRON is on par with SpikeProp. SpikeProp achieves similar classification accuracy with 15 hidden neurons and 2 neurons in the output layer, where as, SEFRON achieves the same with a single neuron.

Ionosphere dataset is another data set that is easily separable. For this data set, the classification accuracy of SEFRON is closer to that of SWAT. SWAT achieves a training accuracy of 86.5% and testing accuracy of 90.0% with 2574 neurons in the hidden layer and 2 neurons in the output layer. On the other hand SEFRON achieves 97.0% training accuracy and 88.9% testing accuracy with a single output neuron.

Based on the obtained results, it may be inferred that the PIMA dataset and the Liver dataset are not easily separable. Yet, the testing accuracy of SEFRON for the PIMA dataset is

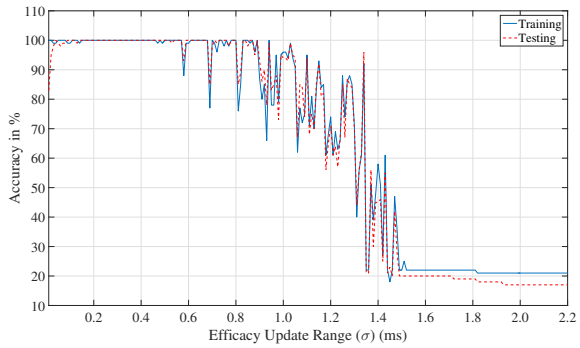


Fig. 4: Effects of σ on the performance of SEFRON

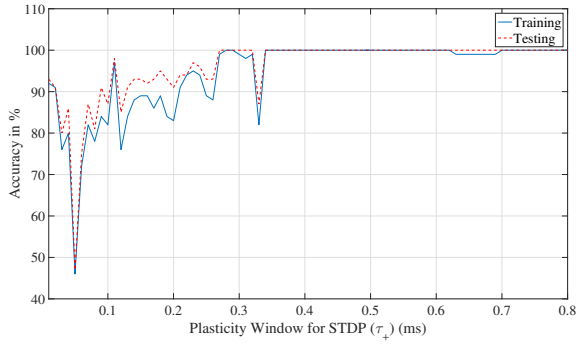


Fig. 5: Effects τ_+ on the performance of SEFRON

closer to that of SpikeProp. However, SpikeProp requires 20 hidden layer neurons to learn the distribution with a 76.2% testing accuracy, whereas SEFRON achieves a 74.0% testing accuracy with only one output neuron. Similar observations can be made also for the Liver dataset.

In summary, a single SEFRON classifier achieves similar performance when compared to other methods which use larger networks. Also, the computational time taken to train a SEFRON classifier is the lowest among all the other methods. The results clearly highlight that SEFRON is computationally more powerful compared to other LIF neuron based networks with constant weights. Hence, replacing the constant weight with a time-varying weight reduces the size of the network and computational time while achieving similar performances.

V. Conclusions

In this paper, a new synapse model with a time-varying synaptic efficacy function incorporated in a LIF neuron, referred to as SEFRON, has been presented. A supervised learning rule for SEFRON is also proposed to approximate the functional relationship between input and output spike patterns. Input-output correlation is encapsulated in the time-varying synaptic efficacy functions by adjusting the weights at different times. SEFRON's learning rule computes the changes in weights (amplitude of the synaptic efficacy function) by minimizing an error function representing the difference in postsynaptic potential due to the fractional contributions of selected presynaptic spikes in a given pattern for both the desired and actual postsynaptic spikes. The resultant synaptic efficacy function can also change continuously from an excitatory to inhibitory nature and this phenomena is similar to the observed GABA-switch phenomenon in a biological neuron.

For binary classification problems, the performance of a single SEFRON has been compared with other well-known spiking neural networks in the literature for four benchmark data sets from the UCI machine learning repository. The results indicate that a single SEFRON captures the classification decision boundary more efficiently and faster than other spiking neural networks with multiple layer/neurons, thereby highlighting the high computational power of a spiking neuron with a time-varying synaptic efficacy function.

VI. Acknowledgment

The authors would like to thank the reviewers for their comments that helped to improve the quality of this paper.

References

[1] W. Maass, "Noisy Spiking Neurons with Temporal Coding have more Computational Power than Sigmoidal Neurons," *Institue of Theoretical Computer Science, Technische Universitaet Graz, Austria, Technical Report*, p. [Online].Available:http://www.igi.tugraz.at/psfile, 1999.

[2] S. M. Bohte, J. N. Kok, and H. La Poutré, "Error-backpropagation in temporally encoded networks of spiking neurons," *Neurocomputing*, vol. 48, pp. 17–37, 2002.

[3] J. J. Wade, L. J. Mcdaid, J. A. Santos, and H. M. Sayers, "SWAT : A Spiking Neural Network Training Algorithm for Classification Problems," *IEEE transactions on neural networks.*, vol. 21, no. 11, pp. 1817–1830, 2010.

[4] F. Ponulak and A. Kasiński , "Supervised Learning in Spiking Neural Networks with ReSuMe : Sequence Learning , Classification , and Spike Shifting," *Neural Computation*, vol. 22, no. 2, pp. 467–510, 2010.

[5] R. Gütig and H. Sompolinsky, "The tempotron: a neuron that learns spike timing–based decisions," *Nature Neuroscience*, vol. 9, no. 3, pp. 420–428, 2006.

[6] A. Mohemmed, S. Schliebs, S. Matsuda, and N. Kasabov, "Span: Spike Pattern Association Neuron for Learning Spatio-Temporal Spike Patterns," *International Journal of Neural Systems*, vol. 22, no. 04, p. 1250012 (17 pages), 2012.

[7] S. Dora, K. Subramanian, S. Suresh, and N. Sundararajan, "Development of a Self-Regulating Evolving Spiking Neural Network for classification problem," *Neurocomputing*, vol. 171, pp. 1216–1229, 2016.

[8] R. V. Florian, "The Chronotron : A Neuron That Learns to Fire Temporally Precise Spike Patterns," *PLoS ONE*, vol. 7, no. 8, p. http://dx.doi.org/10.1371/journal.pone.0040233, 2012.

[9] J. Wang, A. Belatreche, L. Maguire, and T. M. Mcginnity, "An online supervised learning method for spiking neural networks with adaptive structure," *Neurocomputing*, vol. 144, pp. 526–536, 2014.

[10] S. Ghosh-Dastidar and H. Adeli, "A new supervised learning algorithm for multiple spiking neural networks with application in epilepsy and seizure detection," *Neural Networks*, vol. 22, no. 10, pp. 1419–1431, 2009.

[11] O. Booij and H. Nguyen, "A gradient descent rule for spiking neurons emitting multiple spikes," *Information Processing Letters*, vol. 95, pp. 552–558, 2005.

[12] X. Xie, H. Qu, Z. Yi, and J. Kurths, "Efficient Training of Supervised Spiking Neural Network via Accurate Synaptic-Efficiency Adjustment Method," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 6, pp. 1411 – 1424, 2017.

[13] J. Wang, A. Belatreche, L. P. Maguire, and T. M. Mcginnity, "SpikeTemp : An Enhanced Rank-Order-Based Learning Approach for Spiking Neural Networks With Adaptive Structure," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 1, pp. 30–43, 2017.

[14] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *Journal of Physiology*, vol. 117, no. 4, pp. 500–544, 1952.

[15] R. B. Stein, "A Theoretical Analysis of Neuronal Variability," *Biophysical Journal*, vol. 5, no. 2, pp. 173–194, 1965.

[16] R. B. Stein, "Some models of neuronal variability," *Biophysical Journal*, vol. 7, no. 1, pp. 37–68, 1967.

[17] W. M. Kistler, W. Gerstner, and J. Hemmen, "Reduction of the Hodgkin-Huxley Equations to a Single-Variable Threshold Model," *Neural Computation*, vol. 9, no. 5, pp. 1015–1045, 1997.

[18] W. Gerstner, "Time structure of the activity in neural network models," *Physical Review E*, vol. 51, no. 1, pp. 738–758, 1995.

[19] W. Gerstner and W. M. Kistler, *Spiking Neuron Models. Single Neurons, Populations, Plasticity*. Cambridge University Press, 2002.

[20] D. O. Hebb, *The Organization of Behavior; A Neuropsychological Theory*, vol. 63. New York. John Wiley & Sons, 1949.

[21] H. Markram, W. Gerstner, and P. J. Sjöström, *Spike-Timing-Dependent Plasticity: A Comprehensive Overview*. Frontiers Media SA, 2012.

[22] T. V. P. Bliss and T. Lømo, "Long-lasting potentiation of synaptic transmission in the dentate area of the anaesthetized rabbit following stimulation of the perforant path," *The Journal of Physiology*, vol. 232, no. 2, pp. 331–356, 1973.

[23] J. S. Liaw and T. W. Berger, "Dynamic synapses: A new concept of neural representation and computation," *Hippocampus*, vol. 6, no. 1996, pp. 591–600, 1996.

[24] M. V. Tsodyks and H. Markram, "Plasticity of neocortical synapses enables transitions between rates and temporal coding," *Proceedings of ICANN*, pp. 445–450, 1996.

[25] M. Tsodyks and H. Markram, "The neural code between neocortical pyramidal neurons depends on neurotransmitter release probability," *PNAS*, vol. 94, no. 2, pp. 719–723, 1997.

[26] M. Tsodyks, K. Pawelzik, and H. Markram, "Neural networks with dynamic synapses.," *Neural computation*, vol. 10, no. 4, pp. 821–835, 1998.

[27] L. F. Abbott, J. A. Varela, K. Sen, and S. B. Nelson, "Synaptic Depression and Cortical Gain Control," *Science*, vol. 275, no. 5297, pp. 220–224, 1997.

[28] J. S. Dittman, A. C. Kreitzer, and W. G. Regehr, "Interplay between facilitation, depression, and residual calcium at three presynaptic terminals.," *The Journal of neuroscience : the official journal of the Society for Neuroscience*, vol. 20, no. 4, pp. 1374–1385, 2000.

[29] W. Maass and A. M. Zador, "Dynamic stochastic synapses as computational units.," *Neural computation*, vol. 11, no. 4, pp. 903–917, 1999.

[30] A. A. Dibazar, H. H. Namarvar, and T. W. Berger, "A New Approach for Isolated word recognition using dynamic synapse neural networks," *Proceedings of the International Joint Conference on Neural Networks*, vol. 4, pp. 3146–3150, 2003.

[31] J. S. Liaw and T. W. Berger, "Robust speech recognition with dynamic synapses," *1998 IEEE International Joint Conference on Neural Networks Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98CH36227)*, vol. 3, pp. 2175–2179, 1998.

[32] H. H. Namarvar, J. s. Liaw, and T. W. Berger, "A New Dynamic Synapse Neural Network for Speech Recognition," *Neural Networks, 2001. Proceedings. IJCNN '01.*, vol. 4, pp. 2985–2990, 2001.

[33] K. Ganguly, A. F. Schinder, S. T. Wong, and M. M. Poo, "GABA Itself Promotes the Developmental Switch of Neuronal GABAergic Responses from Excitation to Inhibition," *Cell*, vol. 105, no. 4, pp. 521–532, 2001.

[34] S. W. Lee, Y. B. Kim, J. S. Kim, W. B. Kim, Y. S. Kim, H. C. Han, C. S. Colwell, Y. W. Cho, and Y. I. Kim, "GABAergic inhibition is weakened or converted into excitation in the oxytocin and vasopressin neurons of the lactating rat," *Molecular Brain*, vol. 8, no. 1, pp. 1–9, 2015.

[35] S. Suresh, K. Dong, and H. J. Kim, "A sequential learning algorithm for self-adaptive resource allocation network classifier," *Neurocomputing*, vol. 73, no. 16, pp. 3012–3019, 2010.

[36] S. Suresh, S. N. Omkar, V. Mani, and T. N. Guru Prakash, "Lift coefficient prediction at high angle of attack using recurrent neural network," *Aerospace Science and Technology*, vol. 7, no. 8, pp. 595–602, 2003.

Abeegithan Jeyasothy is currently a PhD student in the School of Computer Science and Engineering at Nanyang Technological University. He obtained his B.Eng degree in electrical and electronic engineering from Nanyang Technological University, Singapore. His research interest includes spiking neural networks and machine learning.

Suresh Sundaram (SM'08) received the B.E.degree in electrical and electronics engineering from Bharathiyar University, Coimbatore, India, in 1999, and the M.E. and Ph.D. degrees in aerospace engineering from the Indian Institute of Science, Bengaluru, India, in 2001 and 2005, respectively.

He was a Post-Doctoral Researcher with the School of Electrical Engineering, Nanyang Technological University, Singapore, from 2005 to 2007. From 2007 to 2008, he was with the National Institute for Research in Computer Science and Control, Nice, France, as a Research Fellow of the European Research Consortium for Informatics and Mathematics. He was with Korea University, Seoul, South Korea, for a short period as a Visiting Faculty Member in industrial engineering. He was with the Department of Electrical Engineering, Indian Institute of Technology Delhi, New Delhi, India, as an Assistant Professor, in 2009. Since 2010, he has been an Associate Professor with the School of Computer Science and Engineering, Nanyang Technological University. His current research interests include flight control, unmanned aerial vehicle design, machine learning, and optimization and computer vision.

Narasimhan Sundararajan (LF'11) received the B.E. degree (with First Class Hons.) in electrical engineering from the University of Madras, Chennai, India, in 1966, the M.Tech. degree from the Indian Institute of Technology Madras, Chennai, in 1968, and the Ph.D. degree in electrical engineering from the University of Illinois at Urbana-Champaign, Urbana, IL, USA, in 1971.

From 1971 to 1991, he was researching in different capacities with the Vikram Sarabhai Space Centre, Trivandrum, India, of the Indian Space Research Organization. From 1991, he was a Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore, and retired from that position in 2010. He was a National Research Council Research Associate with NASA Ames Research Center, Ames, CA, USA, in 1974 and a Senior NRC Research Associate with NASA Langley, Hampton, VA, USA, from 1981 to 1986. He is currently a Senior Research Fellow with the School of Computer Engineering, NTU, researching on air traffic management (ATM) research problems. He has published over 250 papers and written six books in the field of computational intelligence and neural networks. His current research interests include ATM, spiking neural networks, neuro-fuzzy systems, and optimization with swarm intelligence.